REMARKS

Claims 1-6, 8-9, 13, 15-16, and 19-20 all the claims pending in the application, stand rejected on prior art grounds. Claims 1, 13, 15-16, and 19-20 stand rejected upon informalities. Claims 1, 13, 15-16, and 19-20 are amended herein. Applicants respectfully traverse the rejections of the claims based on the following discussion.

I. The 35 U.S.C. §112, Second Paragraph, Rejection

Claims 1, 13, 15-16, and 19-20 stand rejected under 35 U.S.C. §112, second paragraph. Page 6 of the Office Action states that claims 1, 13, 15, 16, 19, and 20 all recite "an application comprising a plurality of resource class components" and "respective sets of one or more application instances of each resource class component for the application" and "using hit-weigh[ts] corresponding to a number of hits allocated for each resource instance and an allocated weight for each resource instance". Applicants have amended claims 1, 13, 15, 16, 19, and 20 to recite, in part, "an application comprising a plurality of resource class components <u>comprising tiered layers of web servers, commerce servers, and database servers</u>" and "providing, for each of the application-level users, respective sets of one or more application instances of each resource class component to service the incoming requests from the application-level users to use the application" and have removed the language pertaining to the "hits". In view of the foregoing, the Examiner the use respectfully requested to reconsider and withdraw this rejection.

II. The Prior Art Rejections

Claims 1-3, 5-6, 8-9, 13, 15-16 and 19-20 stand rejected under 35 U.S.C. §102(e) as being anticipated by Abrams, et al. (U.S. Publication No. 2002/0166117A1), hereinafter referred to as "Abrams". Claim 4 stands rejected under 35 U.S.C. §103(a) as being unpatentable over Abrams, in view of Microsoft Computer Dictionary Published in 1997, hereinafter referred to as "Microsoft". Applicants respectfully traverse these rejections based on the following discussion.

Abrams provides on-demand, scalable computational resources to application providers over a distributed network and system. Resources are made available based on demand for applications. Application providers are charged fees based on the amount of resources utilized to satisfy the needs of the application. In providing computer resources, the method and apparatus is capable of rapidly activating a plurality of instances of the applications as demand increases and to halt instances as demand drops. Application providers are charged based on metered amount of computational resources utilized in processing their applications. Application providers access the network to distribute applications onto network to utilize distributed compute resources for processing of the applications. Application providers are further capable of monitoring, updating, and replacing distributed applications. The apparatus and system includes plurality of computing resources distributed across a network capable of restoring and snapshotting provisioned applications based on demand.

Microsoft teaches a definition for "first in, first out" (FIFO) as "a method of processing a queue, in which items are removed in the same order in which they were added - the first in is the first out. Such an order is typical of a list of documents waiting to be printed."
However, the Applicants' independent claims, as amended, include features not taught or suggested by the prior art of record, namely Abrams in view of Microsoft. In particular, claims 1, 15, and 16 generally recite, in part, "...<u>identifying, within a time constraint, failures on any of said multiple networked machines;</u> changing the number of application instances of one or more resource class components in response to the monitored number of requests for each resource

class component <u>and based on machines comprising failures;</u> maintaining a record of the current rate of requests received from respective application-level users, based on the monitored number of serviced requests; and collectively and automatically allocating fractions of different resource class components to a particular application-level user in response to the changed number of application instances of one or more resource class components <u>by using a computational load of each request imposing on said application, wherein said computational load corresponds to a number of requests allocated for each resource instance, wherein said machines comprising failures are prevented from receiving allocations of resources</u>."

Additionally, claims 13, 19, and 20 generally recite, in part, "…<u>identifying, within a time constraint, failures on any of said multiple networked machines;</u> maintaining a record of resources currently available to respective application-level users; and a record of resources currently consumed by respective application-level users; both records of said resources being maintained in respect of each of the one or more application instances of each resource class components; adjusting the respective numbers of said one or more application instances of each resource class component; and <u>collectively and automatically allocating fractions of different resource class components to a particular application-level user in response to a fluctuating number of application instances of one or more resource class components by using a computational load of each request imposing on said application, wherein said computational load corresponds to a number of requests allocated for each resource instance, wherein said machines comprising failures are prevented from receiving allocations of resources, and</u> wherein said application instances of each resource class component are adjusted for each application-level user based (i) at least partly on said records of resources currently available and currently consumed by respective application-level users, (ii) at least partly on predetermined information that estimates the number of each resource class components required to service requests for said application instances of the resource class components, and (iii) at least partly on machines comprising failures.

These features are neither taught nor suggested in Abrams or Microsoft. In particular, Abrams does not teach a process to identify failures on its networked machines (i.e., to detect faulty machines), let alone using this identifying information to determine the number of application instances of one or more resource class components that should be changed/fluctuated. Furthermore, Abrams does not teach that faulty machines (i.e., machines comprising failures) do not receive allocations resources. Abrams briefly describes on page 6, paragraph [0064], "Some of the advantages provided by the on-demand method and system 140 include: protection during peak loads, in one embodiment, with guaranteed application response time SLA; global reach with application provider control of distributed web presence; freedom to grow aggressively including elastic web-processing infrastructure on demand; no capital investment with costs based on the amount of capacity used; supporting substantially any application on substantially any platform to preserve application provider's current application investment; and higher reliability because the system provides superior response time and automatically routes around failures." However, there is no teaching of what constitutes "failures" and whether this routing around failures means that the failures reside on the machines themselves or the applications, or the resource components, and if this means that the failures are fixed or ignored or removed from the system. Also, there is no teaching in Abrams of identifying failures within a particular amount of time (i.e., within a time constraint) (i.e., before a timing out process occurs), which the Applicants' claimed invention provides.

The notion of the virtual server described by Abrams is different from the Applicants' invention. Abrams deals with starting and stopping of application instances depending upon the usage. The virtual server in Abrams is a set of physical servers serving an application for one customer. Whereas, the virtual server provided by the Applicants' invention is defined as a multi-tiered application, which can include multiple instances of each tier (i.e., resource classes). For a customer, the Applicants' invention's virtual server can have multiple instances of a resource class each of which can be hosted on a machine fraction. Thus, the Applicants' invention can allocate a fraction of a machine capacity for each resource class instance and can control/limit usage of resources by that instance which is not the case in Abrams. Accordingly, reference in made to the definition of "virtual server" on page 6 of the specification, as originally filed.

Abrams uses appshot as a technique to increase and decrease the application capacity in response to changing load. Conversely, the Applicants' invention provides a computational load to control the allocation of resources in a fine-grained manner.

The method to handle incoming requests used by Abrams directs all the requests to a single application instance. This is continued until that instance becomes overloaded based on the thresholds defined. Then, the appshot technique is used to start a new instance of the application for handling new incoming requests. Application instances are freed up when the usage goes down below a threshold. Conversely, the Applicants' invention allocates resources to customers based on current load and past usage history (i.e., changed number of application instances of one or more resource class components).

The charging for usage in Abrams is based on the actual hardware resources consumed. Conversely, in the Applicants' invention, a customer is charged based on the number of hits to the virtual server. Moreover, in Applicants' invention, the usage can be specified in terms of the number of requests for use of the application which is a user-friendly parameter as compared to physical resource consumption as used by Abrams.

In view of the foregoing, the Applicants respectfully submit that the cited prior art references, Abrams and Microsoft do not teach or suggest the features defined by amended independent claims 1, 13, 15, 16, 19, and 20 and as such, claims 1, 13, 15, 16, 19, and 20 are patentable over Abrams alone or in combination with Microsoft. Further, dependent claims 2-6, 8, and 9 are similarly patentable over Abrams alone or in combination with Microsoft, not only by virtue of their dependency from patentable independent claims, respectively, but also by virtue of the additional features of the invention they define.

Moreover, the Applicants note that all claims are properly supported in the specification and accompanying drawings. In view of the foregoing, the Examiner is respectfully requested to reconsider and withdraw the rejections.

III. Formal Matters and Conclusion

In view of the foregoing, Applicants submit that claims 1-6, 8-9, 13, 15-16, 19, and 20, all the claims presently pending in the application, are patentably distinct from the prior art of record and are in condition for allowance. The Examiner is respectfully requested to pass the above application to issue at the earliest possible time.

Should the Examiner find the application to be other than in condition for allowance, the Examiner is requested to contact the undersigned at the local telephone number listed below to discuss any other changes deemed necessary. Please charge any deficiencies and credit any overpayments to Attorney's Deposit Account Number 09-0441.

Respectfully submitted,

[illegible]

Dated: November 26, 2005

Mohammad S. Rahman, Esq.
Registration No. 43,029

Gibb I. P. Law Firm, LLC
2568-A Riva Road, Suite 304
Annapolis, MD 21401
Voice: (301) 261-8625
Fax: (301) 261-8825
Customer Number: 29154